Abstract

Phone applications nowadays can show you how many steps you have walked, ran, flights of stairs you have climbed, calories burnt, etc. Online understanding of Human Activities can contribute to some problems existing in the smart city schemes such as healthcare, urban mobility or security. Wearable sensors especially sensors installed in smartphones such as Accelerometer and Gyroscope turn to be very good data streams for Human activity recognition. Smart Phones and other mobile technologies identify their orientation using accelerometer and gyroscope, a small device made up of axis based motion sensing. In this project we intend to classify the physical activities performed by a user based on accelerometer and gyroscope sensor data collected by a smartphone in the user's pocket and neglecting the User bias.

Acknowledgement

It is our privilege to express sincerest regards to our project advisor , Dr.Manimala S for her valuable inputs, able guidance, encouragement, whole-hearted cooperation throughout the duration.

Contents

Chapter 1

Introduction

1.1 Aim

The goal is to classify the heterogenous human activities performed by a user based on accelerometer and gyroscope sensor data collected by a smartphone in the user's pocket by machine learning techniques, considering the data to be streaming data.

1.2 Introduction

Physical activities play a very important role in our physical and mental well being. The lack of physical activities can negatively affect our well-being. Though people know the importance of physical activities, still they need regular motivational feedback to remain active in their daily life. In order to give them proper motivational feedback, we need to recognize their physical activities first. This can be done using heterogeneous sensors already present in Smartphones and SmartWatches prominently accelerometer and Gyroscope. If recognized reliably, this context can enable novel well being applications in different fields for example healthcare.

1.3 Applications

Although the research on activity recognition is beneficial from the mobile sensors un-obtrusiveness, flexibility, and many other advances, it also faces challenges that have brought with them. In this section, review the major, common challenges for activity recognition using mobile sensors, and the corresponding solutions to alleviate them in the current literature.

Subject Sensitivity

The accuracy of activity recognition, especially those based on the accelerometer data, is heavily affected by the subjects participated in training and testing stages. This is mainly due to the fact that different people have different motion patterns. Even for the same subject, she/he may have different patterns at different time.

Location Sensitivity

Due to the property of accelerometer both in wearable sensors and smart phones, its raw reading heavily depends on the sensors orientation and positions on the subjects body. For example, when a user is walking while holding a phone in his/her hand,the moving data reading is quite different from the data reading if the phone is in his/her pocket.

Industry Manufacturing Assisting

The activity recognition techniques could also assist workers in their daily work. This work (wearIT@Work) introduces wearable sensors into work wearable computing is a kind of extension of the body that allows a worker to perform extraordinary tasks.Other applications based on activity recognition include smart cameras that can understand people's gestures in the film shooting field, robot assistance in car production, etc

Localization

Activity recognition on mobile phones could help with context awareness and hence can be applied in localization. One reason to use mobile sensors rather than GPS for localization is that GPS signal is usually very weak inside buildings and underground. On the other hand, the activity recognition techniques with mobile sensors could assist in locating the position. In addition, GPS localization is based positioning which has no information about a users altitude.

Chapter 2

Detailed discussion

2.1 About our Dataset

We have used the “Heterogeneity Human Activity Recognition Dataset” which can be obtained from:

https://archive.ics.uci.edu/ml/datasets/Heterogeneity+Activity+Recognition

It included following information :

The Heterogeneity Dataset for Human Activity Recognition from Smartphone and Smartwatch sensors consists of two datasets (one containing accelerometer readings, the other containing gyroscope readings)

● Number of Attributes : 16 Number of Instances: 26,000,000+

● Users executed activities scripted in no specific order while carrying smartphones.

● Activities: ‘Biking’, ‘Sitting’, ‘Standing’, ‘Walking’, ‘Stairs Up’ and ‘Stairs down’.

● Sensors: Sensors: Two embedded sensors, i.e., Accelerometer and Gyroscope, sampled at the highest frequency the respective device allows (100 Hz)

● Recordings: 9 users and 8 smartphones (2 Samsung Galaxy S3 mini, 2 Samsung Galaxy S3, 2 LG Nexus 4, 2 Samsung Galaxy S+)

2.2 Preprocessing and Data preparation

Why Preprocessing ?

1. Readings from both sensors are relevant in identifying Human Activity but both sensor readings are distributed among 2 different datasets

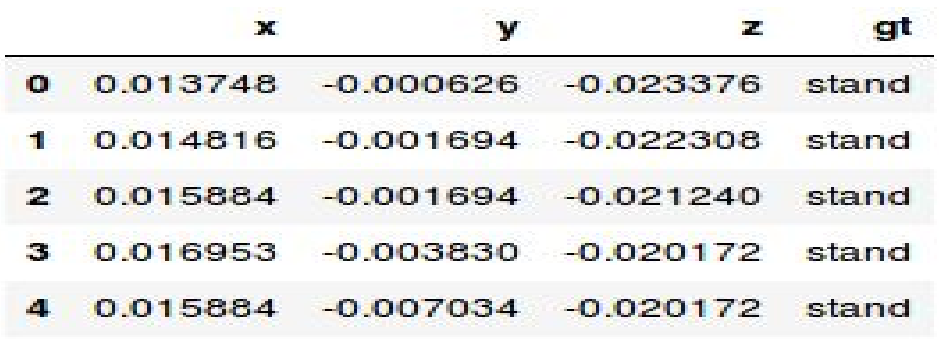
   Hence there is a need to combine these 2 datasets

2. Among 16 features only few features are relevant for recognition of Human Activities.

Relevant features from both Accelerometer and Gyroscope datasets are

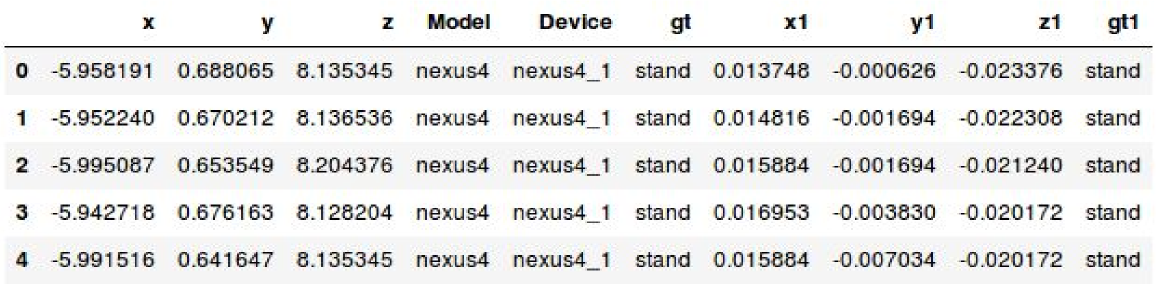
1. Accelerometer
   a. X axis reading
   b. Y axis reading
   c. Z axis reading
   d. Gt - The label for classification task

2. Gyroscope
   a. X axis reading
   b. Y axis reading
   c. Z axis reading
   d. Gt - The label for classification task



	x	y	z	gt
0	0.013748	-0.000626	-0.023376	stand
1	0.014816	-0.001694	-0.022308	stand
2	0.015884	-0.001694	-0.021240	stand
3	0.016953	-0.003830	-0.020172	stand
4	0.015884	-0.007034	-0.020172	stand

- The axis reading features from Gyroscope sensor dataset is merged with the accelerometer dataset.
- The labels of x,y,z,gt from gyroscope is renamed to x1,y1,z1,gt1 before merging to avoid conflicts.



	x	y	z	Model	Device	gt	x1	y1	z1	gt1
0	-5.958191	0.688065	8.135345	nexus4	nexus4_1	stand	0.013748	-0.000626	-0.023376	stand
1	-5.952240	0.670212	8.136536	nexus4	nexus4_1	stand	0.014816	-0.001694	-0.022308	stand
2	-5.995087	0.653549	8.204376	nexus4	nexus4_1	stand	0.015884	-0.001694	-0.021240	stand
3	-5.942718	0.676163	8.128204	nexus4	nexus4_1	stand	0.016953	-0.003830	-0.020172	stand
4	-5.991516	0.641647	8.135345	nexus4	nexus4_1	stand	0.015884	-0.007034	-0.020172	stand

- All the samples with any one reading being is being dropped considering the entire sample as irrelevant or prone to user bias
- The gt1 label is being dropped since it is extended data and similar to gt.
- Since the dataset is very huge (26 million) it was sampled to one fifth by considering one in every five instances.
- The dataset was split to 75% for training purposes and 25% for testing.

Chapter 3

Design and Implementation

3.1 Tools And Technology Used

3.1.1 Python

Python is an interpreted high-level programming language for general-purpose programming.Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community after 30 years.Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation. Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (latebinding), which binds method and variable names during program execution.

- Python is Interpreted.Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- Python is Interactive.You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- Python is Object-Oriented.Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- Python is a Beginner's Language.Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl,Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python's features include:

- Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

- Easy-to-read: Python code is more clearly defined and visible to the eyes.

- Easy-to-maintain: Python's source code is fairly easy-to-maintain.

- A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

- Interactive Mode:Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

- Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

- Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases: Python provides interfaces to all major commercial databases.
- GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable: Python provides a better structure and support for large programs than shell scripting.

Pandas

Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Numpy

NumPy is the fundamental package needed for scientific computing with Python.
This package contains:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- basic linear algebra functions
- basic Fourier transforms
- sophisticated random number capabilities
- tools for integrating Fortran code
- tools for integrating C/C++ code

Besides its obvious scientific uses, *NumPy* can also be used as an efficient multi-dimensional container of generic data. Arbitrary data types can be defined. This allows *NumPy* to seamlessly and speedily integrate with a wide variety of databases.NumPy is a successor for two earlier scientific Python libraries: NumPy derives from the old *Numeric* code base and can be used as a replacement for *Numeric*. It also adds the features introduced by *Numarray* and can also be used to replace *Numarray*.

Keras

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, orTheano. It was developed with a focus on enabling fast experimentation.

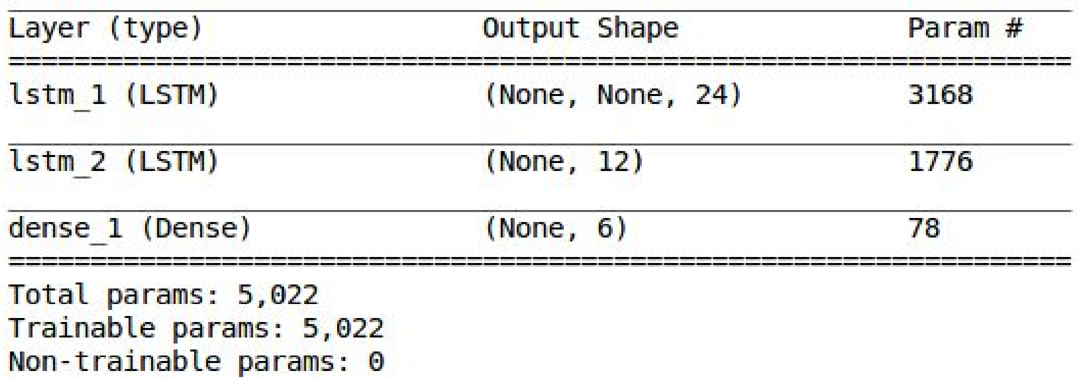
Features of this package :

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

3.2 Design

3.2.1 Classification

- We are implementing a stateless LSTM.
- We added a LSTM layer with 24 neurons, which is followed by another LSTM layer with 12 neurons. This now goes to the final classifying layer, a DENSE layer with 6 neurons whose activation function is sigmoid.
- We have kept Adam optimization as it converges very quickly and it is better than rmsprop.
- Categorical Cross entropy was used as the loss function.
- We have kept Adam optimization as it converges very quickly and it is better than rmsprop.
- Categorical Cross entropy was used as the loss function.



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, None, 24)          3168
_________________________________________________________________
lstm_2 (LSTM)                (None, 12)                1776
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 78
=================================================================
Total params: 5,022
Trainable params: 5,022
Non-trainable params: 0
```

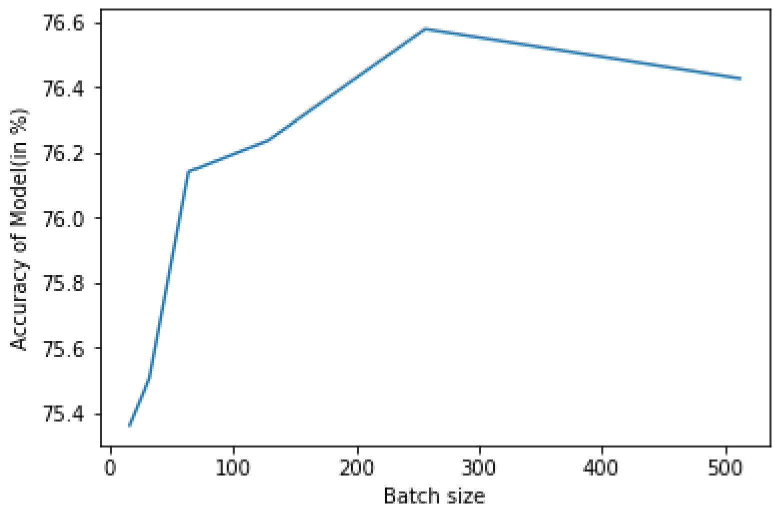
3.2.2 Training

- Our model was trained for 10 epochs for different batch sizes.
- The batch size were in powers of 2 starting from 16 to 1024.
- The test data was evaluated and accuracy was calculated for the same.
- The batch size with highest accuracy is retained.

Chapter 4

Analysis of the Results

The training process was conducted over different batch sizes and cross validated over the testing data. The batch size varied from 16 to 512. The accuracy thus obtained was noted and we have obtained the following curve.



The highest accuracy of 76.57% was obtained for the batch size of 256.

Chapter 5

Conclusion and Future Work

We successfully managed to classify the actions of humans based on sensor data of gyroscope and accelerometer of a smartphone accurately up to a great extent and produced satisfying results using several Machine Learning techniques. But there still remains a lot of scope for improvement. Some of the major improvements which can be done is removing sensor biases and doing a sophisticated feature extraction and to perform the entire classification on larger samples of data. We also intend to work on the data from Heterogenous sensors from Smartwatches as well.

Chapter 6

References

- Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen ”Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition” In Proc. 13th ACM Conference on Embedded Networked Sensor Systems (SenSys 2015), Seoul, Korea, 2015.
- Using Machine Learning on Sensor Data (Journal of Computing and Information Technology - CIT 18, 2010, 4, 341–347 doi:10.2498/cit.1001913)
- https://archive.ics.uci.edu/ml/datasets/Heterogeneity+Activity+Recognition
- https://archive.ics.uci.edu/ml/datasets/Smartphone-Based+Recognition+of+Human+Activities+and+Postural+Transitions